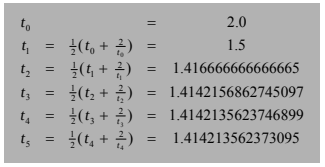
CompSci 6
Programming Design and Analysis



February 4, 2010

Prof. Rodger and Prof. Forbes

Announcements

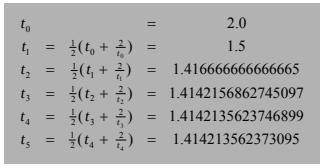
- Reading for next time
  - Chap. 4.6, Chap 7.5, Chap 11.1
  - Reading Quiz due before next class
- Assignment 3 due tonight!
- Assignment 4 out.

Estimation

- Square Root:
  - Given a real number *c* and some error tolerance *epsilon*
  - Estimate *t*, the square root of *c*
- Pi:
  - Estimate π with a given number of *Monte Carlo* trials

While Loops: Square Root

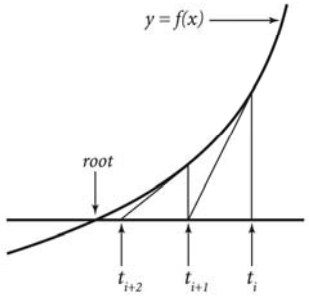
- Q. How might we implement **`Math.sqrt()`**?
- A. To compute the square root of c:
  - Initialize $t_0 = c$.
  - Repeat until $t_i = c / t_i$, up to desired precision: set t_{i+1} to be the average of t_i and c / t_i.



computing the square root of 2

Newton-Raphson Method

- Square root method explained. $f(x) = x^2 - c$ to compute $\sqrt{c}$
  - Goal: find root of function f(x).
  - Start with estimate $t_0 = c$.
  - Draw line tangent to curve at $x = t_i$.
  - Set t_{i+1} to be x-coordinate where line hits x-axis.
  - Repeat until desired precision.



Buffon Needle Experiment

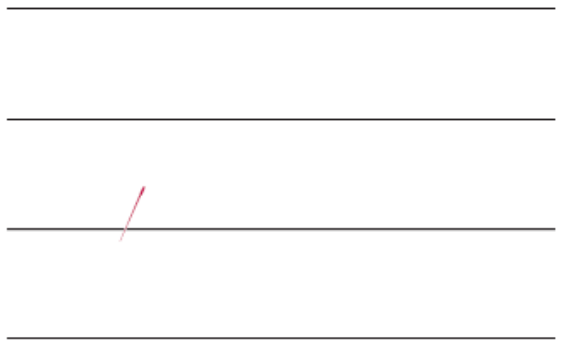


Figure 3 The Buffon Needle Experiment

Needle Position

- Needle length = 1, distance between lines = 2
- Generate random *ylow* between 0 and 2
- Generate random angle α between 0 and 180 degrees
- *yhigh* = *ylow* + sin(α)
- Hit if *yhigh* ≥ 2

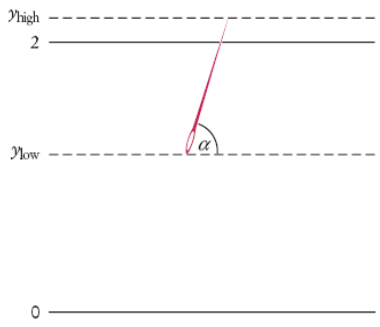


Figure 4
When Does the Needle Fall on a Line?

Constructing objects/Applying methods

- Class Rectangle in Chapter 2
- Creating a Rectangle object with x, y, width, and height

  Rectangle box = new Rectangle(5, 10, 20, 30);
- Applying Methods

```
box.translate(15, 25);      // move the rectangle
System.out.println("x: ", box.getX());   // print x
System.out.println("y: ", box.getY());   // print y
```

Parts of a Class

- State
  - Data
- Constructors
  - Initialize state when object is created
- Accessor methods
  - Accessing data
- Mutator methods
  - Modify data – change the state

Class Example

- Needle class – `Needle.java`
  - Defines state and behavior of Needle
  - Keeps track of the number of times needle hits the line
  - Use `drop()` method to simulate dropping needle
- `java.util.Random` class in Java library
  - `nextDouble()` generates pseudo-random numbers in [0,1]

ch06/random2/Needle.java

```
01: import java.util.Random;
02:
03: /**
04:    This class simulates a needle in the Buffon needle experiment.
05: */
06: public class Needle
07: {
08:    /**
09:       Constructs a needle.
10:    */
11:    public Needle()
12:    {
13:       hits = 0;
14:       tries = 0;
15:       generator = new Random();
16:    }
17:
18:    /**
19:       Drops the needle on the grid of lines and
20:       remembers whether the needle hit a line.
21:    */
```

Continued

ch06/random2/Needle.java (cont.)

```
22:    public void drop()
23:    {
24:       double ylow = 2 * generator.nextDouble();
25:       double angle = 180 * generator.nextDouble();
26:
27:       // Computes high point of needle
28:
29:       double yhigh = ylow + Math.sin(Math.toRadians(angle));
30:       if (yhigh >= 2) myHits++;
31:       tries++;
32:    }
33:
34:    /**
35:       Gets the number of times the needle hit a line.
36:       @return the hit count
37:    */
38:    public int getHits()
39:    {
40:       return myHits;
41:    }
42:
```

Continued

ch06/random2/Needle.java (cont.)

```
43:    /**
44:       Gets the total number of times the needle was dropped.
45:       @return the try count
46:    */
47:    public int getTries()
48:    {
49:       return myTries;
50:    }
51:
52:    private Random myGenerator;
53:    private int myHits;
54:    private int myTries;
55: }
```

Intended Output:

```
Tries = 10000, Tries / Hits = 3.08928
Tries = 1000000, Tries / Hits = 3.14204
```


Classwork Today – Loops/Classes

- Snarf the *classwork* project
- Complete Sqrt
  - Finish *estimate* method
  - Print results
- Complete Needle
  - Finish *main* method
  - Print results
- Classwork handout has all the details
- Submit under assignment name *Class07-Feb04*